International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 7, July - 2013

An IDS (Intrusion Detection System) With Doubleguard

Ambreen Fatima
P.G Student,
Department Of Computer Science and Engineering,
Khaja Banda Nawaz College of Engineering,
Gulbarga, Karnataka, India

Sameena Banu
Assistant Professor,
Department Of Computer Science and Engineering,
Khaja Banda Nawaz College of Engineering,
Gulbarga, Karnataka, India

Abstract

In today's world most of the people uses computer especially for web application to do their transaction. So there are chances of personal data gets hacked then we need to provide more security for both web server and database server. For that double guard system is used. The double guard system is used to detect attacks using Intrusion detection system. An Intrusion Detection System models the network behavior of user sessions across both the front-end web server and the back-end database. By monitoring the web and its subsequent database requests, we are able to ferret out attacks that an independent IDS, would not be able to identify. DoubleGuard is implemented using an Apache web server with MySQL and lightweight virtualization technique. Finally, using DoubleGuard, the wide range of attacks is detected.

Keywords: Virtualization, Intrusion Detection System, Attacks, Container, Session ID.

1. INTRODUCTION

Over a past few years web services and applications had increased in popularity and complexity. As day to day our most of the task such as banking, travel, social networking, and online shopping are done and directly depend on web. The services which are used on the web to run or use the application [8] user interface logic for front end web server which stores the database or file server for particular user data are the back end server. Due to the use of web services which is present everywhere for personal as well as corporate data they have been targeted for the attack. These attacks have recently become more diverse, as attention has shifted from attacking the front-end to exploiting vulnerabilities of the web applications in order to corrupt the back-end database system. A plethora of Intrusion Detection Systems (IDS) currently examine network packets individually within both the web server and the database system. Intrusion detection [9], [11] systems have been widely used to detect the attacks which are known by matching misused traffic patterns or signatures [3], [6] to protect the multi tiered web services. The IDS class has a power of machine learning which can detect unknown attack by identifying the abnormal behavior of the network traffic action from previous behavior of IDS phase. The abnormal network traffic which are send by the attacker to attack the server can be detected by the web IDS and the database IDS [4] and prohibit to enter within the server. But, if the attacker uses the normal traffic to attack the web servers and database server then such type of attack cannot be able to detect by IDSs.

For example, if an attacker with non admin privileges can log into a web server using normal-user access credentials, he/she can find a way to issue a privileged database query by exploiting vulnerabilities in the web server. Neither the web IDS nor the database IDS would detect this type of attack since the web IDS would merely see typical user login traffic and the database IDS would see only the normal traffic of a privileged user. This type of attack can be readily detected if the database IDS can identify that a privileged request from the web server is not associated with user-privileged access. Unfortunately, within the current multi threaded web server architecture, it is not feasible to detect or profile such causal mapping between web server traffic and DB server traffic since traffic cannot be clearly attributed to user sessions.

DoubleGuard is a system which is used to detect the attacks in multitier web services. This approach can create normality model of isolated user sessions which include both the web front-end as HTTP and back-end as File or SQL for network transaction. To achieve this, a lightweight virtualization technique is used for assigning each user's web session to a dedicated container which provides an isolating virtual computing environment. The container ID is used to accurately associate the web request with its subsequent database queries. DoubleGuard will take the web server and database traffic for building a causal mapping profile into proper and accurate account.

2. RELATED WORK

A network Intrusion Detection System can be classified into two types: Anomaly Detection and Misuse Detection.

Anomaly detection first requires the IDS to define and characterize the correct and acceptable static form and dynamic behavior of the system, which can then be used to detect abnormal changes or anomalous behaviors [4], [7]. The boundary between acceptable and anomalous forms of stored code and data is precisely definable. Behavior models are built by performing a statistical analysis on historical data [10], [5], [12] or by using rule-based approaches to specify behavior patterns. An anomaly detector then compares actual usage patterns against established models to identify abnormal events.

In order to detect known web based attacks, misuse detection systems are equipped with a large number of signatures. Unfortunately, it is difficult to keep up with the daily disclosure of web related vulnerabilities, and, in addition, vulnerabilities may be introduced by installation-specific web based applications. Therefore, misuse detection systems should be complemented with anomaly detection systems. An Intrusion Detection System uses a number of different anomaly detection techniques to detect attacks against web servers and web based applications.

In Database Intrusion Detection Using Weighted Sequence Mining, data mining has attracted a lot of attention due to increased generation, transmission and storage of high volume data and an imminent need for extracting useful information and knowledge from them. Database stores valuable information of an application, its security has started getting attention. An Intrusion Detection System (IDS) is used to detect potential violations in database security. In every database, some of the attributes are considered more sensitive to malicious modifications compared to others. The algorithm is proposed for finding dependencies among important data items in a relational database management system. Any transaction that does not follow these dependency rules are identified as malicious.

3. THREAT MODEL AND SYSTEM ARCHITECTURE

The threat model is initially setup to include the assumptions and the types of attacks to protect against. The attackers can bypass the web server to directly attack the database server. Assume that the attacks can neither be detected nor prevented by the current web server IDS, the attackers may take over the web server after the attack, and that afterwards they can obtain full control of the web server to launch subsequent attacks.

3.1 Architecture and Confinement

In the design, a lightweight process container is used, referred to as "containers," as ephemeral, disposable servers for client sessions. It is possible to initialize thousands of containers on a single physical machine, and these virtualized containers can be discarded, reverted, or quickly reinitialized to serve new sessions.

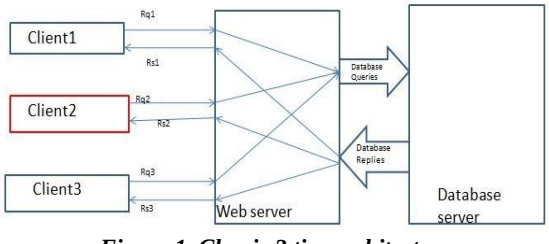

Figure 1. Classic 3 tier architecture

Figure 1 shows the Classic 3 tier architecture. The web server acts as the front end, with the file and database servers as the content storage back end. Database server, unable to know which transaction corresponds to which client request. The communication between the web server and the database server is not separated, and hardly understand the relationships among them. If Client 2 is malicious and takes over the web server, all subsequent database transactions become suspect, as well as the response to the client.

3.2 Building the Normality Model

The container-based and session-separated web server architecture not only enhances the security performances but also provides the isolated

information flows that are separated in each container session. It identifies the mapping between the web server requests and the subsequent DB queries, and utilizes a mapping model to detect abnormal behaviors on a session/client level.

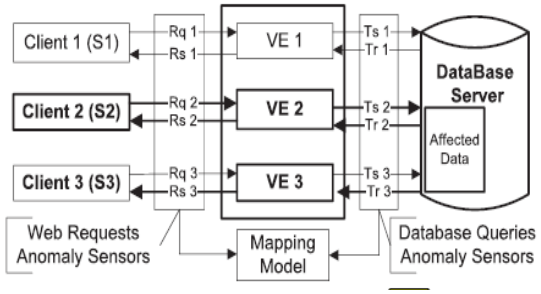

Figure 2. Web server instances running in container

Figure 2 shows how communications are categorized as sessions and how database transactions can be related to a corresponding session. Sensors are used at both sides of the servers. At the web server, sensors are deployed on the host system and cannot be attacked directly since only the virtualized containers are exposed to attackers. Sensors will not be attacked at the database server either, assume that the attacker cannot completely take control of the database server. In fact, assume that sensors cannot be attacked and can always capture correct traffic information at both ends.

Once the mapping model is build, it can be used to detect abnormal behaviors. Both the web request and the database queries within each session should be in accordance with the model. If there exists, any request or query that violates the normality model within a session, then the session will be treated as a possible attack.

3.3 Algorithm

3.3.1 User Control Algorithm

Input: Registration details with username and password as input.

Output: Successful or unsuccessful login.

Algorithm:

1. New user will fill a registration form.
2. Get user name and password.
3. Logs into the system.
4. Starts his new session.
5. After completion of session user logs out.

The above algorithm shows how to provide security to the entire system to prevent unauthorized access of system. If any new user is there, and they want to enter into the system then he/she has to fill a new user registration form. In the registration form user has to fill his personal information along with his username and password. When user clicks on login button all his information get inserted into the database.

Now the user has its own username and password. By clicking on "login" button he/she will redirect to login page. Here user will login into the system by giving his personal username and password. If user enters correct username and password as filled in the registration form; then only the user can successfully login into the system else if the user enters wrong username or password then the user cannot login into the system. Thus this algorithm gives security and provides user control to the system.

3.3.2 Session Handling Algorithm

Input: HTTP request r and SQL query q.

Output: Session id for r and q in the sets ARr and AQq respectively.

Algorithm:

1. for each session separated traffic Ti do
2. Get different HTTP requests 'r' and DB queries 'q' in this session
3. for each different r do
4. if r is a request to static file then
5. Add r into set EQS (Empty Query Set)
6. else
7. if r is not in set REQ then
8. Add r into REQ
9. Append session ID i to the set ARr with r as the key
10. for each different q do
11. if q is not in set SQL then
12. Add q into SQL
13. Append session ID i to the set AQq with q as the key

Session handling algorithm is responsible for assigning correct and unique ID to the HTTP request and equivalent SQL query. If input HTTP request is for any static data/file; means if the requested content is available at web server itself then r is added into Empty Query Set. This type of query doesn't get any kind of ID. If r is not in the set of REQ means the input query is new of arrives first time into the system then r is added into REQ i.e. request query

set. By taking r as a key session ID i is appended to the set of ARr. Similarly for each SQL query if q is not into the set of SQL query then it is added into the SQL set. Same as above by taking q as key session ID i is appended to the set of AQq.

3.3.3 Query Mapping Algorithm

Input: Set of ARr, Set of AQq and Cardinality t.

Output: HTTP request gets mapped with equivalent SQL query.

Algorithm:

1. for each distinct HTTP request r in REQ do
2. for each distinct DB query q in SQL do
3. Compare the set ARr with the set AQq
4. if ARr =AQq and Cardinality(ARr) > t then
5. Found a Deterministic mapping from r to q
6. Add q into mapping model set MSr of r
7. Mark q in set SQL
8. else
9. Need more training sessions
10. return False
11. for each DB query q in SQL do
12. if q is not marked then
13. Add q into set NMR (No Matched Request)
14. for each HTTP request r in REQ do
15. if r has no deterministic mapping model then
16. Add r into set EQS (Empty Query Set)
17. return True

The user request comes to the web server in the form of HTTP request and a equivalent SQL query is generated by web server. Query mapping algorithm maps the HTTP request with the equivalent SQL query. Mapping algorithm use the output generated by the session handling algorithm. A HTTP request with its ID stored in ARr set and a SQL query with its ID stored in AQq set; both are matched with each other if both ID are equal and Cardinality of ARr is greater than 1 then there is a deterministic map is found. q is then added into the matched set query and it is also marked in the set of SQL queries. After performing all training data sets if any query from the set q is not marked then that q is moved to the NMR (No Matched Request) set. Similarly for every HTTP request r; if r has no deterministic mapping then that r is added into the EQS (Empty Query Set).

3.3.4 Intrusion Detection Algorithm

Input: HTTP request r and SQL query q.

Output: Log showing malicious attacks.

Algorithm:

1. If the rule for the request is Deterministic Mapping r -> Q (Q ≠Φ), test whether Q is a subset of a query set of the session. If so, this request is valid then mark the queries in Q. Otherwise, a violation is detected and considered to be abnormal and the session will be marked as suspicious.
2. If the rule is Empty Query Set r -> Φ, then the request is not considered to be abnormal and it will not mark any database queries. No intrusion will be reported.
3. For the remaining unmarked database queries, see if they are in the set NMR. If so, mark the query as such.
4. Any untested web request or unmarked database query is considered to be abnormal. If either exists within a session, then that session will be marked as suspicious.

The intrusion detection algorithm checks every r and q with the mapping model and then decides that whether it is from a general user or attacker. If there is mapping found between r and q then it is a considered as valid session, otherwise it have to checks other query sets. If request r is found in Empty Query Set then it not considered as abnormal and no intrusion will be reported. For remaining unmark queries, see if they are in the set NMR. If so, mark the query as such. Any query that comes directly to the database without any mapping then that session is considered as abnormal.

4. PERFORMANCE EVALUATION

A prototype of DoubleGuard is implemented using a web server with a back-end DB and setup the dynamic websites. To evaluate the detection results for the system, four classes of attacks is analyzed.

4.1 Implementation

In this prototype, each user session is assigned into a different container. In the implementation, containers were recycled based on events. The same session tracking mechanisms is used as implemented by the Apache server (cookies, mod user track, etc.) because lightweight virtualization containers do not impose high memory and storage overhead.

4.2 Dynamic Model Detection Rate

The model building is conducted for the dynamic blog website.

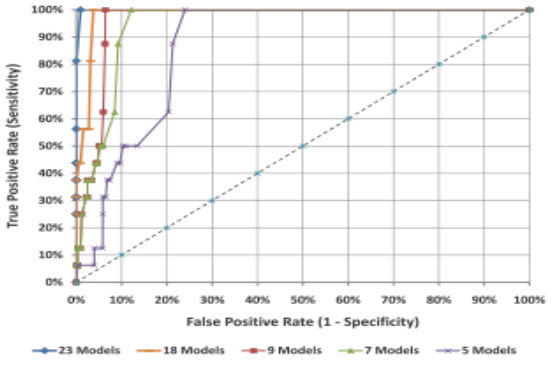

Figure 3. False positive rate for dynamic models

Figure 3 shows the ROC curves for the testing result. The model is built with different number of operations, and each point on the curves indicates the threshold value. The threshold value is defined as the number of HTTP requests or SQL queries in a session that are not matched with the normality model. The threshold value is varied from 0 to 30 during the detection. As the ROC curves depict, it could always achieve a 100 percent True Positive Rate when doing strict detection (threshold of 0) against attacks in the threat model. With a more accurate model, it can reach 100 percent sensitivity with a lower False Positive rate. The nature of the false positives comes from the fact that manually extracted basic operations are not sufficient to cover all legitimate user behaviors. Figure 3 show that it can reach 100 percent Sensitivity with six percent False Positive rate.

5. ATTACK DETECTION

DoubleGuard is used to detect the malicious attacks. It uses the attack tools listed in Table 1 and also shows the experiment views for DG.

5.1 Privilege Escalation Attack

This type of attack is actually done by accessing privilege of authorized user by unauthorized users. Suppose there is an application for the Payment System for Employee's in which Administrator privilege to update and change the salary of the employee has and employee have privilege to see their attendance. If any employee gets the URL to update the salary then he/she gets the access of all the employee salary. In case, the attacker employee will get the privilege of the admin and

Table 1.
Detection Results for Attacks (GSQL Stands for GreenSQL, and DG Stands for DoubleGuard, *Indicates Attack Using Metasploit)

Operation	Snort	GSQL	DG
Privilege Escalation (WordPress Vul)	No	No	Yes
Web Server aimed attack (nikto)	Yes	No	Yes
SQL Injection (sqlmap)	No	Yes	Yes
DirectDB	No	No	Yes
linux/http/ddwrt_cgibin_exec*	No	No	Yes
linux/http/linksys_apply_cgi*	No	No	Yes
linux/http/piranha_passwd_exec*	No	No	Yes
unix/webapp/oracle_vm_agent_utl*	No	No	Yes
unix/webapp/php_include*	Yes	No	Yes
unix/webapp/php_wordpress_lastpost*	No	No	Yes
windows/http/altn_webadmin*	No	No	Yes
windows/http/apache_modjk_overflow *	No	No	Yes
windows/http/oracle9i_xdb_pass*	No	No	Yes
windows/http/maxdb_webdbm_database*	No	No	Yes

privilege escalation attack is done.

If the Payment System uses the DoubleGuard application then it will be placed after the DG. DG will store the admin privilege and employee privilege separately in the DG database. Whenever the admin or employee want to use the Payment System application then they has to go from DG's privilege authentication where according to the user i.e. admin or employee and its privilege the DG application will take to their respective privilege pages according to the user register privileges in the DoubleGuard database. DG will never show the URL of the respective application database. In this way, DG will prevent privilege attack.

5.2 Hijack Future Session Attack

Whenever the internet services or application through web browser is used, it generates a unique session ID and it remains until or task is not completed or web browser is closed. Attacker tries to get this session ID. So that attacker can get the valuable data and it's most common examples are FACEBOOK, GMAIL etc. After getting session ID the attacker can do anything he wants with the user data. But the original user doesn't know that attacker is accessing his/her data which would turn harmful for the user.

If the user uses the DG application he will be prevented from such kind of attack. In DG

application, the Mapping Model for the session ID and IP address is created. If the attacker will be able to get the session ID then also it will not possible to him/her to attack the user data because the IP address of the attacker will not match with DG's Mapping Model. DG will allow the access if the session ID and IP address are match according to the mapping model of application database. Depending upon the result of the DG it will decide the user is legal or not and allow him/her access the database or not.

5.3 SQL Injection Attack

Now-a-days the attackers are using the SQL queries to get the data or change the data of another user by sending queries like INSERT, UPDATE, DELETE, etc. In this kind of attack, the attacker communicates with the database by sending queries. But whiles ending the SQL queries by an attacker the structure of the queries are changed and which are never detected by the IDS. But, the DG application is able to prevent the injection attack because the DG will generate its own structure queries and which are different from the attacker SQL queries structure. DG will allows to access, update the database if structure of the SQL queries are matched with the structure of the DG application query structure.

5.4 Direct DB Attack

Most of the attacker directly attacks the database server besides going to the web server. In this kind of attacking, the attacker uses the IP address of the database server. It is very easy and less time requirement attack. In this attacker sends the SQL queries directly to the database server by bypassing the web server. If the DG is used then the attack will be detected and attacker will not be allowed to the database server. If DG is used then it will be placed before the web server and the database server. So that, DG will be able to hide the IP address and location where the database server is located and DG doesn't match the web request with the SQL queries. Thus DG can avoid such kind of attacks.

6. CONCLUSION

In this paper, an Intrusion Detection System builds the normality model for multitier web applications. Unlike previous approaches that correlated or summarized alerts generated by independent IDSs, DoubleGuard forms container-based IDS with multiple input streams to produce alerts. The lightweight virtualization technique is used to assign session ID to a dedicated container which is nothing but isolated virtual computing environment. Furthermore, there will specific detection of attacks such as Privilege Escalation Attack, Hijack Future Session Attack, SQL Injection Attack and Direct DB Attack. Also the requests which violate the normality model that will be treat as an intruder. DoubleGuard was able to identify a wide range of attacks with minimal false positives. DoubleGuard is used for dynamic web server which provides better security for data and web application.

7. FUTURE SCOPE

It is possible to make some future modifications into the Intrusion Detection System. The Intrusion Detection Systems can be installing on wide range of machines having different operating system and platforms. The query processing mechanism can be made simpler by applying natural language processing (NLP); so as to convert simple English sentences into SQL queries. New attacks are often unrecognizable by popular IDS. So there is continuous race going in between new attacks and detection systems have been a challenge. Nowadays Intrusion Detection Systems also work on the wireless networks. The latest wireless devices come with its own set of protocols for communication that break the traditional OSI layer model. So IDS must learn new communication patterns of the latest wireless technology.

8. REFERENCES

[1] P. Vogt, F. Nentwich, N. Jovanovic, E. Kirda, C. Kru¨ gel, and G. Vigna, "Cross Site Scripting Prevention with Dynamic Data Tainting and Static Analysis," Proc. Network and Distributed System Security Symp. (NDSS '07), 2007.

[2] V. Felmetsger, L. Cavedon, C. Kruegel, and G. Vigna, "Toward Automated Detection of Logic Vulnerabilities in Web Applications," Proc. USENIX Security Symp., 2010.

[3] B.I.A. Barry and H.A. Chan, "Syntax, and Semantics-Based Signature Database for Hybrid Intrusion Detection Systems," Security and Comm. Networks, vol. 2, no. 6, pp. 457-475, 2009.

[4] H. Debar, M. Dacier, and A. Wespi, "Towards a Taxonomy of Intrusion-Detection Systems," Computer Networks, vol. 31, no. 9, pp. 805-822, 1999.

[5] G. Vigna, W.K. Robertson, V. Kher, and R.A. Kemmerer, "A Stateful Intrusion Detection System for World-Wide Web Servers," Proc. Ann. Computer Security Applications Conf. (ACSAC '03), 2003.

[6] J. Newsome, B. Karp, and D.X. Song, "Polygraph: Automatically Generating Signatures for Polymorphic Worms," Proc. IEEE Symp. Security and Privacy, 2005.

[7] T. Verwoerd and R. Hunt, "Intrusion Detection Techniques and Approaches," Computer Comm., vol. 25, no. 15, pp. 1356-1365, 2002.

[8] S. Potter and J. Nieh, "Apiary: Easy-to-Use Desktop Application Fault Containment on Commodity Operating Systems," Proc. USENIX Ann. Technical Conf., 2010.

[9] A. Srivastava, S. Sural, and A.K. Majumdar, "Database Intrusion Detection Using Weighted Sequence Mining," J. Computers, vol. 1, no. 4, pp. 8-17, 2006.

[10] C. Kruegel and G. Vigna, "Anomaly Detection of Web-Based Attacks," Proc. 10th ACM Conf. Computer and Comm. Security (CCS '03), Oct. 2003.

[11] F. Valeur, G. Vigna, C. Krugel, and R.A. Kemmerer, "A Comprehensive Approach to Intrusion Detection Alert Correlation," IEEE Trans. Dependable and Secure Computing, vol. 1, no. 3, pp. 146-169, July-Sept. 2004.

[12] M. Cova, D. Balzarotti, V. Felmetsger, and G. Vigna, "Swaddler: An Approach for the Anomaly-Based Detection of State Violations in Web Applications," Proc. Int'l Symp. Recent Advances in Intrusion Detection (RAID '07), 2007.

[13] "Wordpress," http://www.wordpress.org/, 2011.

[14] nikto, http://cirt.net/nikto2, 2011.

[15]sqlmap,http://sqlmap.sourceforge.net/, 2011.